

Energy in computing systems with speed scaling: optimization and mechanisms design

Oscar C. Vásquez*

Abstract

We study a simple scheduling game for the speed scaling model. Players want their job to complete early, which however generates a big energy consumption. We address the game from the mechanism design side, and by charging the energy usage to the players we seek for a good compromise between quality of service and energy usage.

1 Introduction

It is common knowledge that we entered now a period of humanity where natural resources become rare. This situation triggered consciousness in responsible consumption, and in particular many countries, companies and individuals aim in minimizing their energy consumption. Minimizing this resources is a relatively new topic in decision theory, and opened to new problems and research areas.

Usually minimizing energy consumption is an opposed goal to maximizing quality of service, think for example at transport systems. In this paper we consider the particular case of computing systems with speed scaling. We study a simplified model, with a unique processor, that can run at a variable continuous and unbounded speed. Users submit jobs to this machine, each job has some workload-representing a number of instructions to execute. Then clearly running these jobs at high speed will be benefic for the users, since it leads to small job completion time which is the quality of service experienced by the users. On the other side running at low speed, will be benefic for the machine, which is then more energy efficient.

In the presence of two opposed goals there is the need to combine them into a single objective value, which we call the *social welfare*. For this work we decided to represent the energy consumption cost and the quality of service as some utility value. The machine is controlled by a single *efficient regulator* who maximizes the total social welfare, deciding on the speed and the order in which jobs are to be scheduled on the machine.

2 General Model

Formally, we consider a non-cooperative game with n users and a regulator. The regulator manages the machine where the jobs are executed. Each user has a job i with a workload w_i

*LIP6, Université Pierre et Marie Curie, Paris, France and Engineering Industrial Department, University of Santiago of Chile.

and a quality of service function Q_i . Since the regulator can observe the workload of job i , once it completes, it makes sense to assume that he knows w_i in advance.

The regulator states some *cost sharing mechanism* C , which is announced publicly. This mechanism defines a cost share function C_i to be charged to user i , which is supposed to compensate the energy consumption cost generated by the execution of the workload w_i ; and specifies the information $\{\hat{I}_i\}$ requested by the regulator from user i in order to compute an optimum schedule s for a specific objective. Note that the announced information \hat{I}_i and the real information I_i are not necessarily the same.

In this context a schedule is simply defined by two objects: a speed-function s , mapping every time point to a non-negative speed, as well as an ordering of the jobs. We denote this order by a permutation σ such that $\sigma(i)$ is the rank of job i . For convenience, we denote by π the inverse of σ , i.e. $\pi(j)$ is the j -th job to be scheduled. In the sequel we identify a schedule by the speed function s and an ordering of the jobs.

The schedule defines a completion time t_i for each job i and generates an energy consumption cost defined as $E(s) := \int_0^{+\infty} s^\alpha(t)dt$, and $\int_0^{t_{\pi(j)}} s(t)dt = \sum_{k=1}^j w_{\pi(k)}$ for all j , for some constant α which is usually assumed to be $2 \leq \alpha \leq 3$.

Finally, the regulator charges a cost share C_i and announces a completion time \hat{t}_i for each user i , which could be different from t_i . By normalization we can assume that the welfare function for the cost share C_i is the identity, i.e. the value of cost share in monetary unit is the same value in utility unit.

The objective of each user i is to minimize his welfare

$$W_i := Q_i(\hat{t}_i) - C_i(s),$$

whereas the objective of the regulator is to maximize the total welfare.

$$WT := \sum_{i=1}^n W_i.$$

For a given mechanism, these values depend on the information \hat{I} declared by the users, so we might adopt a function notation for WT .

In this model, the strategy for user i is an information \hat{I}_i that he has to announce. We call \hat{I}'_i as an *improvement* for the user i if his welfare W_i for the strategy profile $(\hat{I}_{-i}, \hat{I}'_i)$ is strictly larger than for the strategy profile $\hat{I} = (\hat{I}_{-i}, \hat{I}_i)$. Here $(\hat{I}_{-i}, \hat{I}'_i)$ stands for the strategy profile which is identical with \hat{I} , except at index i , where the strategy for player i is \hat{I}'_i . Player i is happy in the strategy profile \hat{I} if there is no possible *improvement* for him. A strategy profile I , where every player is happy, is called a *pure Nash equilibrium*. As we do not mention mixed Nash equilibria in this work, we will omit the term *pure* from now on.

With these definitions in mind, we are facing two related problems:

The Centralized optimization problem where the regulator has an objective to maximize and the necessary information for computing an optimum is known to him.

The Mechanism design problem where the regulator has an objective to maximize and the necessary information for its computing is only known to the users. In this situation, the regulator must design a cost sharing mechanism such that each user will be happy if he announces his private information as required by the regulator.

2.1 Desirable properties of cost sharing mechanisms

We have the game theoretical problem of designing a cost sharing mechanism C with the following desired properties.

1. The mechanism should be *budget-balanced*, meaning that $\sum_i C_i(s) = E(s)$. If not, we want at least a β -*budget-balanced* mechanism for some constant β , meaning that $E(s) \leq \sum_i C_i(s) \leq \beta E(s)$.
2. The mechanism C should always admit a Nash equilibrium. In particular, the mechanism will be *strategy proof* if the strategy profile, where every user announces his private information, is a Nash equilibrium.
3. The mechanism C cannot arbitrarily exclude any users; meaning that if user i is willing to pay enough (at least C_i) then his job is executed.
4. In the mechanism C , every user has the possibility to exclude himself, resulting that his job won't be executed, the user won't be charged and experience an individual welfare of zero.

2.2 Users

We study two types of users according, depending on their welfare functions for quality of service, which can either be Q_i^A and Q_i^B as defined below. Both functions are monotone non-increasing in the completion time of job \hat{t}_i , and translating how important it for user i that his job i completes early. Type A users have some constant utility when the job completes before some private deadline and zero utility if it completes late. Type B users have some utility which is linear with the job completion time. So there is a given utility if their job completes at the ideal time zero, and the utility fades linearly with increasing completion time. The formal definitions of these quality of service functions are given below in equations 1 and 2 for some constants $U_i^A, d_i, U_i^B, p_i > 0$.

$$Q_i^A(\hat{t}_i) := \begin{cases} U_i^A & \text{if } \hat{t}_i \leq d_i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$Q_i^B(\hat{t}_i) := U_i^B - p_i \hat{t}_i \quad (2)$$

The variants of optimization and mechanism problems are detailed in the next sections.

3 Type A users

In this section we consider the case where all users are of type A. This means that every user has a quality of service function Q_i^A using a private value U_i^A and private deadline d_i . The regulator requests the private deadlines from the users.

3.1 Optimization problem

We consider the centralized optimization problem consisting in the maximization of the total welfare. Suppose that the regulator knows the deadlines d of all users and the sum of utility

values $\sum_i U_i^A$ is sufficiently large, such that the maximum total welfare is when all users participate to the game. By considering a budget balanced mechanism, we have:

Theorem 1. *Consider a game with type A users. Assume that total welfare is maximized when all users participate to the game. The problem of maximizing the total welfare is equivalent to the problem of minimizing the consumed energy cost which can be computed in time $O(n \log n)$.*

Minimizing the consumed energy cost has been widely studied, in a more general setting with release times. The paper [7] provided the first polynomial time algorithm which time complexity was improved from $O(n^3)$ to $O(n^2 \log n)$ by [3]. This has been improved to $O(n^2)$ if the release time–deadline intervals are laminar, i.e. have a tree structure [2].

In [7] it was observed by an averaging argument that in an optimal schedule s , if job i is scheduled at speed $s(t)$ at some time t , then in the whole span of the job — which is $[0, d_i)$ in our case — the speed must be at least $s(t)$. From this fact follows that in an optimal schedule the speed function s is non-increasing and even piecewise constant. So we can describe the schedule by a sequence of pairs of the form (v, ℓ) where v/ℓ stands for a speed, ℓ for the duration of an interval and v for the workload which can be done in this interval ℓ . We use this argument in our proof.

Proof. (sketch) Assume that jobs are ordered according to deadlines, i.e. $d_1 \leq \dots \leq d_n$. By an exchange argument without loss of generality jobs are scheduled by an optimal schedule in the same order. This means that in our case a schedule is really defined by the speed function s . The optimization problem can be formulated as

$$\max \sum_{i=1}^n W_i = \max \sum_{i=1}^n Q_i^A(s) - E(s) = \max \sum_{i=1}^n U_i^A - E(s) \quad (3)$$

s.t

$$\int_0^{d_i} s(t) dt \geq \sum_{j=1}^i w_j \quad \forall i. \quad (4)$$

Since $\sum_{i=1}^n U_i^A$ is constant, the same schedule minimizes $E(s)$ under condition (4). Thus, the mechanism needs to compute a speed function s minimizing the consumed energy while respecting all deadlines.

We describe now a linear time algorithm to compute the optimal schedule s , assuming jobs are already ordered according to deadlines. Let O_i be a stack of pairs in decreasing order of speed, describing the optimal schedule for job set $\{1, \dots, i\}$. We define O_0 as the empty stack. Now O_i is obtained from O_{i-1} in the following manner. First we push on O_{i-1} the pair $(w_i, d_i - d_{i-1})$. Then while O_{i-1} contains at least two pairs, and the two top pair (v, ℓ) and second top pair (v', ℓ') satisfy $v'\ell \leq v\ell'$, we replace them by $(v + v', \ell + \ell')$. The invariant is that $\sum_{(v, \ell) \in O_i} \ell = d_i$ and the pair (v, ℓ) are in strict decreasing order of $\frac{v}{\ell}$. The proof correctness is omitted.

Thus, our specific problem can be solved in time $O(n \log n)$, the necessary time to sort jobs according to deadlines. \square

3.2 Mechanism design problem

We adopt the above assumptions and define that the information requested by the regulator are the deadlines d of the jobs. This allows him to optimize total welfare. Thus, only if the information announced by users are the true private values, the regulator is able to optimize total welfare.

We study the *proportional cost sharing* mechanism. Its definition is quite easy, every user is charged exactly the cost generated by the workload of its job. In this mechanism, the regulator announces the true job completion times, that is $\hat{t}_i = t_i$.

Theorem 2. *The game with type A users and the proportional cost sharing mechanism is strategy proof.*

Proof. Being strategy proof means that if each player announces the true individual deadlines, i.e. $\hat{d}_i = d_i$ then the resulting strategy profile is a Nash equilibrium. Let $C_i^p(s)$ be the cost share for player i , as defined by the *proportional cost sharing mechanism* C^p . We define $C_{\pi(j)}^p(s) := (t_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$, where for convenience we abuse notation and write $t_{\pi(0)} := 0$.

We assume that each user $\pi(j)$ wants to maximize his welfare $W_{\pi(j)}(s)$. The mechanism satisfies $\hat{t}_{\pi(j)} = t_{\pi(j)} \leq \hat{d}_{\pi(j)}$. No player knows the information of the other players. We show that for this situation announcing the true value is a dominant strategy, in the sense that it leads to a pure Nash equilibrium. The idea is similar to second-price sealed-bid auction. Table 1 shows the payoff of user $\pi(j)$ in function his strategy and the regulator's strategy.

Table 1: Payoff for different strategies in different cases

		User's strategy		
		$\hat{d}_{\pi(j)} < d_{\pi(j)}$	$\hat{d}_{\pi(j)} = d_{\pi(j)}$	$\hat{d}_{\pi(j)} > d_{\pi(j)}$
Regulator's strategy	$t_{\pi(j)} < \hat{d}_{\pi(j)}$	$(t_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$	$(t_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$	$(t_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$ or 0
	$t_{\pi(j)} = \hat{d}_{\pi(j)}$	$(\hat{d}_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$	$(d_{\pi(j)} - t_{\pi(j-1)})^{1-\alpha} w_{\pi(j)}^\alpha$	0

It is easy to see that the strategy $\hat{d}_{\pi(j)} = d_{\pi(j)}$ is dominant. \square

In order to measure the proportional cost sharing mechanism, we see that it is *budget-balanced* and it does not exclude any users by definition. In addition, if we assume that each user i has a non negative welfare when his workload is executed and the cost share is charged, then this mechanism will be *efficient*. This latter fact implies uniqueness of the Nash equilibrium which also optimizes total welfare.

4 Type B users

In this section we consider the game with type B users. Every player i has now a quality of service function Q_i^B . In this game the regulator needs to receive from the players the workloads w and deadlines d of the jobs, so he can maximize total welfare.

4.1 Optimization problem

We assume that the regulator knows the penalties p of the jobs and the sum of utility values $\sum_i U_i^B$ is sufficiently large, such that the total welfare is maximized when all users participate in the game. We consider a budget balanced mechanism, and we obtain the following result, which has been independently obtained by Megow and Verschae [4].

Theorem 3. *Consider a game with type B users. Assume that the total welfare is maximized when all users participate in the game. The problem of maximizing the total welfare is equivalent to the classical scheduling problem denoted as 1|| $\sum w_i t_i^\beta$ for $\beta := (\alpha - 1)/\alpha$.*

Proof. Fix some optimal schedule. The jobs complete in some order π , and by the j -th job we refer to this order. For any $j > 1$ the j -th job schedules in the interval $[t_{\pi(j-1)}, t_{\pi(j)})$ whereas the first job schedules in the interval $[0, t_{\pi(1)})$. Denote by ℓ_j the length of the execution interval of the j -th job. From the averaging argument from [7] we know that the jobs are scheduled throughout their interval at equal speed $s_j := w_{\pi(j)}/\ell_j$. With these notations the completion time of the j -th job can be written as $\sum_{i=1}^j \ell_i$.

Thus, the optimization problem can be formulated as:

$$\max \sum_{i=1}^n W_i(s) = \max \sum_{i=1}^n Q_i^B(s) - E(s) = \max \sum_{i=1}^n (U_i^B - p_i t_i) - E(s). \quad (5)$$

s.t.

$$\int_0^{t_j} s(t) dt \geq \sum_{i=1}^j w_i \quad \forall j. \quad (6)$$

Again, since $\sum_{i=1}^n U_i^A$ is constant, the same schedule maximizes the following expression under condition (6):

$$\min \sum_{i=1}^n p_i t_i + E(s). \quad (7)$$

And, by definition of $s(t)$, we have:

$$\min \sum_{j=1}^n w_{\pi(j)}^\alpha \ell_j^{1-\alpha} + \sum_{j=1}^n p_{\pi(j)} \sum_{i=1}^j \ell_i. \quad (8)$$

The first order condition of this function on the variable ℓ_j for an arbitrary index j states

$$(1 - \alpha) w_{\pi(j)}^\alpha \ell_j^{-\alpha} + \sum_{k=j}^n p_{\pi(k)} = 0,$$

which implies

$$w_{\pi(j)}^\alpha \ell_j^{-\alpha} = \frac{\sum_{k=j}^n p_{\pi(k)}}{\alpha - 1}.$$

Replacing these equalities in the social cost value leads to

$$\begin{aligned} & \sum_{j=1}^n \ell_j \left(\frac{\sum_{k=j}^n p_{\pi(k)}}{\alpha - 1} + \sum_{k=j}^n p_{\pi(k)} \right) \\ &= \frac{\alpha}{\alpha - 1} \sum_{j=1}^n \ell_j \sum_{k=j}^n p_{\pi(k)} \end{aligned}$$

The first order condition on ℓ_j above gave the equality

$$\ell_j = \left(\frac{w_{\pi(j)}^\alpha (\alpha - 1)}{\sum_{k=j}^n p_{\pi(k)}} \right)^{\frac{1}{\alpha}}$$

which permits to simplify as

$$\begin{aligned} & \frac{\alpha}{\alpha - 1} \sum_{j=1}^n \left(\frac{w_{\pi(j)}^\alpha (\alpha - 1)}{\sum_{k=j}^n p_{\pi(k)}} \right)^{\frac{1}{\alpha}} \sum_{k=j}^n p_{\pi(k)} \\ &= \alpha (\alpha - 1)^{\frac{1-\alpha}{\alpha}} \sum_{j=1}^n w_{\pi(j)} \left(\sum_{k=j}^n p_{\pi(k)} \right)^{\frac{\alpha-1}{\alpha}}. \end{aligned}$$

Now consider the following scheduling problem, which also consists of n jobs, but this time, the processing time of job i is p_i , its priority weight is w_i and its completion denoted by C_i . The aim is to schedule these jobs on a single machine, with unit speed, so to minimize

$$\sum_i w_i C_i^{\frac{\alpha-1}{\alpha}}.$$

Clearly every optimal schedule to the above problem corresponds to an schedule maximizing total welfare in the former model, just by reversing the job order. In conclusion, the optimization problem is polynomial equivalent to a classical scheduling problem denoted as $1||\sum w_i C_i^\beta$ for $\beta := (\alpha - 1)/\alpha$. \square

The complexity of this problem type remains still open. Concerning approximation algorithms, Stiller and Wiese [6] show that the Smith's rule [5] guarantees an approximation factor of $(\sqrt{3} + 1)/2 \approx 1.366$ which is tight when f is part of the problem input, whereas Höhn and Jacobs [1] derive a simple method to compute the tight approximation factor of a Smith-ratio-schedule for any particular monotone increasing convex or concave cost function.

4.2 Mechanism design problem

From the previous theorem it follows that in order to maximize total welfare, the game regulator needs to know the individual penalties p of the players. Therefore we aim for a truthful cost sharing mechanism. To fix the notations, we call X the mechanism proposed in this section, and assume that the game regulator announces completion times \hat{t}_i which are the true completion times t_i of the schedule produced by the regulator.

We define now the cost sharing mechanism X , where $C_i^x(s)$ is the cost share for player i . The mechanism computes the schedule that maximizes total welfare. By the j -th job, we refer to the order in which jobs complete therein. We define the cost share for the j -th player $\pi(j)$ as follows:

$$C_{\pi(j)}^x(s) := \sum_{k=1}^j (\alpha s_k^\alpha - \hat{p}_{\pi(j)}) \ell_k$$

where

$$s_k^\alpha = \frac{\sum_{r=k}^n \hat{p}_r}{\alpha - 1} = \frac{w_k^\alpha}{\ell_k^\alpha}.$$

Note that $C_{\pi(j)}^x(s)$ is equivalent to:

$$(\alpha - 1)^{-1} \sum_{k=1}^j \left(\hat{p}_j + \alpha \sum_{r=k, r \neq j}^n \hat{p}_r \right) \ell_k$$

where

$$\ell_k = \left(\frac{w_k \sum_{r=k}^n \hat{p}_r}{\alpha - 1} \right)^{-1/\alpha}.$$

Hence, the cost share for the first player in the sequence is the penalty weighted execution time over all other jobs plus its energy cost.

Theorem 4. *The game with type B players that the cost sharing mechanism X is strategyproof.*

Proof. Consider the definition of $C_i^x(s)$. We assume that each player $\pi(j)$ want to maximize his welfare $W_{\pi(j)}(s)$. We show that PNE is obtained as an local minimum of $W_{\pi(j)}(s)$. By definition, he have that the player i has a welfare is

$$U_i^B - p_i * t_i - (\alpha - 1)^{-1} \sum_{k=1}^{\sigma(i)} (\hat{p}_i + \alpha \sum_{r=k, r \neq \sigma(i)}^n \hat{p}_r) \ell_k$$

We now analyze when a changing unilateral of strategy \hat{p}_i does't improve the welfare. We consider the first derive of welfare function.

$$\frac{\partial W_i}{\partial \hat{p}_i} = 0$$

Thus, we have:

$$\begin{aligned}
0 &= p_i \sum_{k=1}^{\sigma(i)} \frac{\delta \ell_k}{\delta \widehat{p}_i} + \sum_{k=1}^{\sigma(i)} (\alpha \frac{\delta s_k^\alpha}{\delta \widehat{p}_i} - 1) \ell_k + (\alpha s_k^\alpha - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i} \\
&\quad \sum_{k=1}^{\sigma(i)} (p_i - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i} + \sum_{k=1}^{\sigma(i)} \alpha \frac{\delta s_k^\alpha}{\delta \widehat{p}_i} \ell_k - \ell_k + \alpha s_k^\alpha \frac{\ell_k}{(1-\alpha)\alpha s_k^\alpha} \\
&\quad \sum_{k=1}^{\sigma(i)} (p_i - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i} + \sum_{k=1}^{\sigma(i)} \alpha \frac{\delta s_k^\alpha}{\delta \widehat{p}_i} \ell_k - \ell_k + \frac{\alpha \ell_k}{(1-\alpha)\alpha} \\
&\quad \sum_{k=1}^{\sigma(i)} (p_i - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i} + \sum_{k=1}^{\sigma(i)} \frac{\alpha}{\alpha-1} \ell_k - \ell_k - \frac{\ell_k}{\alpha-1} \\
&\quad \sum_{k=1}^{\sigma(i)} (p_i - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i} + \sum_{k=1}^{\sigma(i)} \ell_k \frac{\alpha - \alpha + 1 - 1}{\alpha - 1} \\
&\quad \sum_{k=1}^{\sigma(i)} (p_i - \widehat{p}_i) \frac{\delta \ell_k}{\delta \widehat{p}_i}
\end{aligned}$$

Given that $\frac{\delta \ell_k}{\delta \widehat{p}_i} \neq 0$, then \widehat{p}_i must be p_i . □

References

- [1] Wiebke Höhn and Tobias Jacobs. On the performance of Smith's rule in single-machine scheduling with nonlinear cost. In *Proceedings of the 10th Latin American Theoretical Informatics Symposium (LATIN '12)*, LNCS, 2012. To appear.
- [2] Minming Li, Becky Jie Liu, and Frances F. Yao. Min-energy voltage allocation for tree-structured tasks. *J. Comb. Optim.*, 11(3):305–319, 2006.
- [3] Minming Li, Andrew C. Yao, and Frances F. Yao. Discrete and continuous min-energy schedules for variable voltage processor. *Proceedings of the National Academy of Sciences of the United States of America*, 103(11):3983–3987, 2006.
- [4] Nicole Megow and José Verschae. Scheduling on a machine with varying speed: Minimizing cost and energy via dual schedules. Technical report, arxiv.org, 2012.
- [5] W.E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956.
- [6] S. Stiller and A. Wiese. Increasing speed scheduling and flow scheduling. *Algorithms and Computation*, pages 279–290, 2010.
- [7] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science, FOCS '95*, pages 374–382, Washington, DC, USA, 1995. IEEE Computer Society.